### LAMBDA FUNCTION -> dashQuestionDel\_R

API URL ->https://j0mmgihtaj.execute-api.us-east-1.amazonaws.com/v1/dashQuestionDel\_R

Resource -> [**dashquestiondel**](https://us-east-1.console.aws.amazon.com/apigateway/main/apis/7efwp1v3ed/resources?api=7efwp1v3ed&region=us-east-1)

Stage -> v1

### 1. Role of the API

This API deletes a quiz question (either MCQ or descriptive) from the database based on the provided questionId.

### 2. Functioning

* **Token Verification**: Validates the JWT token to authenticate the request.
* **CORS Handling**: Sets appropriate headers to handle Cross-Origin Resource Sharing (CORS).
* **Database Connection**: Ensures a consistent MongoDB connection.
* **Request Validation**: Ensures required fields like questionId are present and valid.
* **Question Deletion**: Deletes the quiz question (either MCQ or descriptive) based on the provided questionId.
* **Response Handling**: Returns success or error messages based on the operation's outcome

### 3. Request Body

**Headers**:

Key: Authorization

Value: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiI2NjQwNTFlM2M0N2EwMmVkNjBlNDU3MTAiLCJlbWFpbCI6InNoaXZhbmlzaW5naDAzMTIwMkBnbWFpbC5jb20iLCJpYXQiOjE3MTg0NzA2MjYsImV4cCI6MTcxODQ4ODYyNn0.jZCiZFauL79bcE\_3kOjwR0x0JWmgGfPAaJfkzednPx8

**Body**:

{

"questionId": "666dc70a29513fe8f56fa81b"

}

### 4. Response

**Success Response:**

* **Status Code**: 200

{

"statusCode": 404,

"headers": {

"Access-Control-Allow-Origin": "\*",

"Access-Control-Allow-Headers": "Content-Type, Authorization",

"Access-Control-Allow-Methods": "OPTIONS,POST,GET"

},

"body": "{\"message\":\"Question not found\"}"

}

**Error Responses:**

* **Status Code**: 401
  + **Body**: {"message": "No token provided"} | {"message": "Invalid token"}
* **Status Code**: 404
  + **Body**: {"message": "Question not found"}
* **Status Code**: 500
  + **Body**: {"message": "Internal Server Error"}

### 5. Logic

1. **Token Verification**: Validates the JWT token to ensure the user is authorized to perform the operation.
2. **CORS Headers**: Sets headers to allow CORS requests from any origin.
3. **Database Connection**: Ensures a consistent connection to MongoDB using Mongoose.
4. **Request Validation**: Ensures required fields like questionId are present and valid.
5. **Question Deletion**: Deletes the quiz question (either MCQ or descriptive) based on the provided questionId.
6. **Response Creation**: Generates and returns appropriate responses based on the success or failure of the operation.

### 6. Dependencies

* **mongoose**: MongoDB object modeling tool.
* **jsonwebtoken**: Library for generating and verifying JSON Web Tokens (JWT).

CODE:

const mongoose = require('mongoose');

const jwt = require('jsonwebtoken');

const Quiz = require('./question'); // Adjust the path as needed

// MongoDB connection URI

const MONGO\_URI = process.env.MONGODB\_URI;

// JWT secret key

const JWT\_SECRET = process.env.JWT\_SECRET\_KEY;

// Initialize MongoDB connection

mongoose.connect(MONGO\_URI);

exports.handler = async (event) => {

// Handle preflight request

if (event.httpMethod === 'OPTIONS') {

return {

statusCode: 204,

headers: {

"Access-Control-Allow-Origin": "https://admin.exambuilder.online",

"Access-Control-Allow-Headers": "Content-Type, Authorization",

"Access-Control-Allow-Methods": "OPTIONS,POST,GET"

},

body: null

};

}

try {

console.log('Event:', JSON.stringify(event, null, 2)); // Log the incoming event

// Check if Authorization header exists

const token = event.headers.Authorization || event.headers.authorization;

if (!token) {

return {

statusCode: 401,

headers: {

"Access-Control-Allow-Origin": "https://admin.exambuilder.online"

},

body: JSON.stringify({ message: 'No token provided' })

};

}

// Verify JWT token

try {

jwt.verify(token, JWT\_SECRET);

} catch (error) {

return {

statusCode: 401,

headers: {

"Access-Control-Allow-Origin": "https://admin.exambuilder.online"

},

body: JSON.stringify({ message: 'Invalid token' })

};

}

// Parse questionId from request body

let body;

try {

body = JSON.parse(event.body);

} catch (parseError) {

console.error('JSON Parse Error:', parseError);

return {

statusCode: 400,

headers: {

"Access-Control-Allow-Origin": "https://admin.exambuilder.online"

},

body: JSON.stringify({ message: 'Invalid JSON payload' })

};

}

const { questionId } = body;

// Validate questionId

if (!mongoose.Types.ObjectId.isValid(questionId)) {

return {

statusCode: 400,

headers: {

"Access-Control-Allow-Origin": "https://admin.exambuilder.online"

},

body: JSON.stringify({ message: 'Invalid question ID' })

};

}

// Find the quiz document containing the question

const quiz = await Quiz.findOne({

$or: [

{ 'mcqQuizz.\_id': questionId },

{ 'descriptiveQuizz.\_id': questionId }

]

});

if (!quiz) {

return {

statusCode: 404,

headers: {

"Access-Control-Allow-Origin": "https://admin.exambuilder.online"

},

body: JSON.stringify({ message: 'Question not found' })

};

}

// Determine question type and find the index

let questionType;

let questionIndex;

if (quiz.mcqQuizz.some(mcq => mcq.\_id.toString() === questionId)) {

questionType = 'mcqQuizz';

questionIndex = quiz.mcqQuizz.findIndex(mcq => mcq.\_id.toString() === questionId);

} else if (quiz.descriptiveQuizz.some(desc => desc.\_id.toString() === questionId)) {

questionType = 'descriptiveQuizz';

questionIndex = quiz.descriptiveQuizz.findIndex(desc => desc.\_id.toString() === questionId);

} else {

return {

statusCode: 404,

headers: {

"Access-Control-Allow-Origin": "https://admin.exambuilder.online"

},

body: JSON.stringify({ message: 'Question not found' })

};

}

// Create a copy of the question

const questionCopy = quiz[questionType][questionIndex].toObject();

// Retain the original \_id in the copy

// Save the question copy to quiztrash collection

const QuizTrash = mongoose.connection.collection('quiztrash');

await QuizTrash.insertOne(questionCopy);

// Store the question ID in preVersionID, ensuring existing IDs are not deleted

if (!quiz.preVersionID.includes(questionId)) {

quiz.preVersionID.push(questionId);

}

// Remove the question from mcqQuizz or descriptiveQuizz

quiz[questionType].splice(questionIndex, 1);

await quiz.save();

return {

statusCode: 200,

headers: {

"Access-Control-Allow-Origin": "https://admin.exambuilder.online"

},

body: JSON.stringify({ message: 'Question deleted successfully' })

};

} catch (error) {

console.error('Error:', error);

let statusCode = 500;

let errorMessage = 'Internal Server Error';

if (error instanceof mongoose.Error.ValidationError) {

statusCode = 400;

errorMessage = 'Validation Error';

} else if (error.name === 'JsonWebTokenError') {

statusCode = 400;

errorMessage = 'Invalid Token';

}

return {

statusCode,

headers: {

"Access-Control-Allow-Origin": "https://admin.exambuilder.online"

},

body: JSON.stringify({ message: errorMessage })

};

}

};